Transmitting Recorded Material

The present invention is concerned with methods and apparatus for transmitting recorded material, such as video, audio of other material to be played in real time, over a network.

According to one aspect of the present invention, there is provided a method of transmitting a recording comprising:

- commencing transmission thereof;
- holding received data in a receiver buffer; and
- commencing playing of said received data;

characterised by the steps of analysing the whole of the recording to determine a point at which to commence playing such that no buffer underflow can occur; and commencing playing only when this point has been reached.

In another aspect, the invention provides a method of transmitting a recording comprising:

- commencing transmission thereof;
- holding received data in a receiver buffer; and
- commencing playing of said received data;

characterised by the steps of:

analysing the whole of the recording to identify a first section at the beginning thereof which meets the condition that it covers a playing time interval greater than or equal to the maximum of the timing error for a following section of any length, each timing error being defined as the extent to which the transmission time of the respective following section exceeds its playing time interval; and causing the receiver to commencing playing only after said first section has been received.

Further aspects of the invention are set out in the claims

Some embodiments of the invention will now be described, by way of example, with reference to the accompanying drawings, in which:

Figure 1 is a block diagram of a transmission system embodying the invention;

Figure 2 is a timing diagram;

Figure 3 is a flowchart explaining the operation of the control unit shown in Figure 1;

Figure 4 is a flowchart explaining an alternative mode of operation of the control unit; and

Figure 5 is a flowchart explaining a yet further version.

In Figure 1, a streamer 1 contains (or has access to) a store 11 in which are stored files each being a compressed version of a video sequence, encoded using a conventional compression algorithm such as that defined in the ITU standard H.261 or H.263, or one of the ISO MPEG standards. Naturally one may store similar recordings of further video sequences, but this is not important to the principles of operation.

By "bit-rate" here is meant the bit-rate generated by the original encoder and consumed by the ultimate decoder; in general this is not the same as the rate at which the streamer actually transmits, which will be referred to as the transmitting bit-rate. It should also be noted that these files are generated at a variable bit-rate (VBR) - that is, the number of bits generated for any particular frame of the video depends on the picture content. Consequently, references above to low (etc.) bit-rate refer to the average bit-rate.

The server has a transmitter 12 which serves to output data via a network 2 to a terminal 3. The transmitter is conventional, perhaps operating with a well known protocol such as TCP/IP. A control unit 13 serves in conventional manner to receive requests from the terminal for delivery of a particular sequence, and to read packets of data from the store 11 for sending to the transmitter 12 as and when the transmitter is able to receive them. Here it is assumed that the data are read out as discrete packets, often one packet per frame of video, though the possibility of generating more than one packet for a single frame is not excluded. (Whilst is in principle possible for a single packet to contain data for more than one frame, this is not usually of much interest in practice).

Note that these packets are not necessarily related to any packet structure used on the network 2.

The terminal 3 has a receiver 31, a buffer 32, and a decoder 33.

Some networks (including TCP/IP networks) have the characteristic that the available transmitting data rate fluctuates according to the degree of loading on the network.

Some theoretical discussion is in order at this point.

As shown in Figure 2, an encoded video sequence consists of N packets. Each packet has a header containing a time index t_i (i=0 ... N-1) (in terms of real display time – e.g. this could be the video frame number) and contains b_i bits. This analysis assumes that packet i must be completely received before it can be decoded (i.e. one must buffer the whole packet first).

In a simple case, each packet corresponds to one frame, and the time-stamps t_i increase monotonically, that is, $t_{i+1} > t_i$ for all i. If however a frame can give rise to two or more packets (each with the same t_i) then $t_{i+1} \geq t_i$. If frames can run out of capture-and-display sequence (as in MPEG) then the t_i do not increase monotonically. Also, in practice, some frames may be dropped, so that there will be no frame for a particular value of t_i.

These times are relative. Suppose the receiver has received packet 0 and starts decoding packet 0 at time $t_{ref} + t_0$. At "time now" of $t_{ref} + t_g$ the receiver has received packet t_g (and possibly more packets too) and has just started to decode packet g.

Packets g to h-1 are in the buffer. Note that (in the simple case) if $h = g + 1$ then the buffer contains packet g only. At time $t_{ref} + t_j$ the decoder is required to start decoding packet j. Therefore, at that time $t_{ref} + t_j$ the decoder will need to have received all packets up to and including packet j.

The time available from now up to $t_{ref} + t_j$ is $(t_{ref} + t_j) - (t_{ref} + t_g) = t_j - t_g$. (1)

The data to be sent in that time are that for packets h to j, viz.

$$\sum_{i=h}^{j} b_i \qquad (2)$$

which at a transmitting rate R will require a transmission duration

$$\frac{\sum_{i=h}^{j} b_i}{R} \qquad (3)$$

This is possible only if this transmission duration is less than or equal to the time available, i.e. when the currently available transmitting rate R satisfies the inequality

$$\frac{\sum_{i=h}^{j} b_i}{R} \leq t_j - t_g \qquad (4)$$

Note that this is the condition for satisfactory reception and decoding of packet *j*: satisfactory transmission of the whole of the remaining sequence requires that this condition be satisfied for all $j = h \ldots N\text{-}1$.

For reasons that will become apparent, we rewrite Equation (4) as:

$$\frac{\sum_{i=h}^{j} b_i}{R} - (t_j - t_{h-1}) \leq t_{h-1} - t_g \qquad (5)$$

Note that $t_j - t_{h-1} = \sum_{i=h}^{j} (t_i - t_{i-1}) = \sum_{i=h}^{j} \Delta t_i$ where $\Delta t_i = t_i - t_{i-1}$.

Also, we define $\Delta\varepsilon_i = (b_i / R) - \Delta t_i$

Note that $t_{h-1} - t_g$ is the difference between the time-stamp of the most recently received packet in the buffer and the time stamp of the least recently received packet in the buffer – i.e. the one that we have just started to decode.

Then the condition is

$$\sum_{i=h}^{j} \Delta\varepsilon_i \leq t_{h-1} - t_g \qquad (6)$$

For a successful transmission up to the last packet *N-1*, this condition must be satisfied for any possible *j*, viz.

$$Max_{j=h}^{j=N-1} \left\{ \sum_{i=h}^{j} \Delta\varepsilon_i \right\} \leq t_{h-1} - t_g \qquad (7)$$

The left-hand side of Equation (7) represents the maximum timing error that may occur from the transmission of packet *h* up to the end of the sequence, and the condition states, in effect that this error must not exceed the ability of the receiver buffer to accommodate it, given its current contents. For convenience, we will label the left-hand side of Equation (7) as T_h - i.e.

$$T_h = Max_{j=h}^{j=N-1} \left\{ \sum_{i=h}^{j} \Delta\varepsilon_i \right\} \qquad (8)$$

So that Equation (7) may be written as

$$T_h \leq t_{h-1} - t_g \qquad (9)$$

Consider the situation at time $t_g = t_0$, that is, when the decoder is to commence decoding of the first packet. In the general case, the above condition will not be satisfied when there is only one packet in the buffer (h=1). The receiver waits for the buffer contents to reach a satisfactory level before it commenced decoding. Using the above condition, it becomes apparent that the receiver should wait at least until the buffer contains packet H-1 where H is the smallest value of h for which the condition

$$T_h \leq t_{h-1} - t_0 \tag{10}$$

is satisfied.

In this embodiment of the invention, one of the functions of the control unit 13 is that, each time it sends a packet to the transmitter 12, it evaluates the test embodied in Equation 10.

Figure 3 is a flowchart showing operation of the control unit. At step 101 a packet counter is reset. Then (102) the first packet (or on subsequent iterations, the next packet) is read from the store 11 and sent to the transmitter 12. At step 103, the control unit computes the value of T_n. At this point, the counter n points to the last packet sent, whereas Equation (10) is formulated for the last packet sent being h-1. Consequently the calculation at step 103 is of T_{n+1} and the test performed at step 104 is whether $T_{n+1} \leq t_n - t_0$.

If this test is not passed, the packet counter is incremented at 106 and control returns to step 102 where, as soon as the transmitter is ready to accept it, a further packet is read out and transmitted. If the test is passed, then it is known that the receiver is safe to begin decoding as soon as it has received this packet. Therefore at step 105 the control unit sends to the transmitter a "start" message to be sent to the receiver. When the receiver receives this start message, it begins decoding. If there is any possibility of messages being received in a different order from that in which they were sent, then the start message should contain the packet index n so that the receiver may check that packet n has actually been received before it commences decoding. Alternatively, the transmitter could send values of T_{n+1} to the receiver, and the receiver itself performs the test.

Following the sending of the "start" message, the packet counter is incremented at 107 and another frame transmitted at 108: these steps are repeated until the end of the file is reached, this being recognised at 109 and the process terminates at 110.

The preceding description assumes that the control unit performs this calculation each time it sends a packet to the transmitter, which is computationally quite intensive. An alternative is to perform the calculation less often, perhaps once every five packets, which reduces the amount of computation but may result in the buffering of more frames than is necessary.

Another alternative is to complete the computation as soon as it is able to do so (i.e. without waiting for the next packet) and then send a start message (with starting packet number) to the receiver. A yet further alternative is to perform the computation before transmitting any packets at all. Once the value of h is determined, we then transmit packets 0 to h-1 in reverse order (packet $h-1$, packet $h-2$, ... packet 0). In this case it ceases to be necessary to transmit an explicit "start" command. Standard receivers that support UDP transport protocol are able to reorder packets, and will automatically wait until packet 0 has arrived before commencing decoding. In fact, it is sufficient that packet 0 is withheld until after packets 1 to $h-1$ have been sent (whose order is immaterial).

This however precludes the possibility of taking into account changes in the transmitting data rate R during the waiting period, and is therefore satisfactory only if such changes are not expected.

Observe (by inspection of Equation (3)) that the significance of the rate R is in calculating the time taken to send packets h to j. Therefore the actual rate used to transmit packets 0 to $h-1$ is of no consequence as it does not affect the result.

Another attractive option is to perform as much as possible of the computation in advance. If a system in which only one value of R is possible, or permitted, then the computation of T_{n+1} at step 103 and the test of step 104 can be performed in advance for each frame up to the point where the test is passed, and the result recorded in the file, for example by recording the corresponding value of n in a separate field at the start of the file, or by attaching a special flag to frame n itself. Thus in Figure 3, steps 103 and 104 would be replaced by the test "is current value of n equal to the value of n stored in the file?"; or "does current frame contain the start flag?". Alternatively the separate field (or flag) could be forwarded to the receiver and this recognition process performed at the receiving end.

Figure 4 shows a flowchart of a process for dealing with the situation where the transmitting data rate R varies. In principle this involves T_h for every packet and storing this value in the packet header. In practice however it is necessary to compute them

for a sufficient number of frames (perhaps 250 frames at 25 frames per second) at the beginning of the sequence that one is confident that the test will be passed within this period. Unfortunately, the calculation of T_h involves the value of R, which is of course unknown at the time of this pre-processing. Therefore we proceed by calculating T_h for a selection of possible values of R, for example (if R_A is the average bit rate of the file in question)

$$R_1 = 0.5R_A$$
$$R_2 = 0.7R_A$$
$$R_3 = R_A$$
$$R_4 = 1.3R_A$$
$$R_5 = 2R_A$$

So each packet h has these five precalculated values of T_h stored in it. If required (for the purposes to be discussed below) one may also store the relative time position at which the maximum in Equation (8)) occurs, that is,

$\Delta t_{h\max} = t_{j\max} - t_h$ where $t_{j\max}$ is the value of j in Equation 8 for which T_h is obtained.

In this case the flowchart proceeds as follows following transmission of frame n:

112: interrogate the transmitter 12 to determine the available transmitting rate R;

103A:EITHER - in the event that R corresponds to one of the rates for which T_h has been precalculated - read this value from the store;

OR - in the event that R does not so correspond, read from the store the value of T_h (and, if required, $t_{h\,max}$) that correspond to the highest one (R^-) of the rates $R_1 \ldots R_5$ that is less than the actual value of R, and estimate T_h from it;

104A: Apply the test $T_{n+1} + \Delta \le t_n - t_0$, where Δ is a fixed safety margin;

Continue as before.

The estimate of T_h could be performed simply by using the value T_h^- associated with R^-; this would work, but since it would overestimate T_h it would result, at times, in the receiver waiting longer than necessary. Another option would be by linear (or other) interpolation between the values of T_h stored for the two values of $R_1 \ldots R_5$ each side of the actual value R. However, our preferred approach is to calculate an estimate according to:

$$T_i' = \frac{(T_i^- + \Delta t^-_{i\max})R^-}{R} - \Delta t^-_{i\max} \qquad (11)$$

Where R^- is the highest one of the rates $R_1 \ldots R_5$ that is less than the actual value of R, T_i^- is the precalculated T_h for this rate, $\Delta t^-_{i\max}$ is the time from t_i at which T_i^- is obtained (i.e. is the accompanying value of $\Delta t^-_{h\max}$). In the event that this method returns a negative value, we set it to zero.

Note that this is only an estimate, as T_h is a nonlinear function of rate. However with this method T_i' is always higher than the true value and automatically provides a safety margin (so that the margin Δ shown above may be omitted).

Note that these equations are valid for the situation where the encoding process generates two or more packets (with equal t_i) for one frame, and for the situation encountered in MPEG with bidirectional prediction where the frames are transmitted in the order in which they need to be decoded, rather than in order of ascending t_i.

We will now describe an alternative embodiment in which the mathematics is converted into an equivalent form which however, rather than performing the calculations for each packet individually, makes use of calculations already made for a preceding packet. Recalling Equation(8):

$$T_h = Max_{j=h}^{j=N-1}\left\{\sum_{i=h}^{j}\Delta\varepsilon_i\right\}$$

which may be rewritten

$$T_h = Max\left\{\sum_{i=h}^{h}\Delta\varepsilon_i, Max_{j=h+1}^{j=N-1}\left\{\sum_{i=h}^{h}\Delta\varepsilon_i + \sum_{i=h+1}^{j}\Delta\varepsilon_i\right\}\right\} \qquad (12)$$

$$= Max\left\{\Delta\varepsilon_h, Max_{j=h+1}^{j=N-1}\left\{\Delta\varepsilon_h + \sum_{i=h+1}^{j}\Delta\varepsilon_i\right\}\right\}$$

$$= \Delta\varepsilon_h + Max\left\{0, Max_{j=h+1}^{j=N-1}\left\{\sum_{i=h+1}^{j}\Delta\varepsilon_i\right\}\right\}$$

$$= \Delta\varepsilon_h + Max\{0, T_{h+1}\} \qquad (13)$$

Provided that $T_{h+1} \geq 0$, which will be true at the beginning of the file, this becomes

$$T_h = T_{h+1} + \Delta\varepsilon_h \tag{14}$$

Or generally

$$T_{a+1} = T_a - \Delta\varepsilon_a$$

$$T_{a+2} = T_{a+1} - \Delta\varepsilon_{a+1} = T_a - \Delta\varepsilon_a - \Delta\varepsilon_{a+1}$$

$$T_{a+b} = T_a - \sum_{i=a}^{a+b-1} \Delta\varepsilon_i$$

If b=h-a then

$$T_h = T_a - \sum_{i=a}^{h-1} \Delta\varepsilon_i \tag{15}$$

substituting $\Delta\varepsilon_i = \frac{b_i}{R} - \Delta t_i$ and $\Delta t_i = t_i - t_{i-1}$

$$T_h = T_a - \sum_{i=a}^{h-1} \frac{b_i}{R} + \sum_{i=a}^{h-1} \Delta t_i = T_a - \sum_{i=a}^{h-1} \frac{b_i}{R} + (t_{h-1} - t_{a-1}) \tag{16}$$

If a=0, then

$$T_h = T_0 - \sum_{i=0}^{h-1} \frac{b_i}{R} + (t_{h-1} - t_{-1})$$

If a=1, then

$$T_h = T_1 - \sum_{i=1}^{h-1} \frac{b_i}{R} + (t_{h-1} - t_0)$$

Consider the test

$$T_h \leq t_{h-1} - t_0$$

which may be written

$$T_a - \sum_{i=a}^{h-1} \frac{b_i}{R} + (t_{h-1} - t_0) \leq t_{a-1} - t_0 \tag{17}$$

if a=0, this becomes

$$T_0 - \sum_{i=0}^{h-1} \frac{b_i}{R} \leq +t_{-1} - t_0 \tag{18}$$

Noting that t_{-1} is a meaningless quantity (appearing on both sides on the inequality) so that it can be given any value, it is convenient to define t_{-1} as equal to t_0, whence we obtain

$$T_0 \le \sum_{i=0}^{h-1} \frac{b_i}{R} \qquad (19)$$

(or, if $a = 1$: $T_1 \le \sum_{i=1}^{h-1} \frac{b_i}{R}$)

Thus the test of Equation (10)

$$T_h \le t_{h-1} - t_0$$

could instead be written

$$T_0 \le \sum_{i=0}^{h-1} \frac{b_i}{R} \qquad (20)$$

Then the first test (h=1) is Test 1:

$$T_0 \le \frac{b_0}{R} ?$$

Or, if we define $Z_x = T_0 - \frac{1}{R}\sum_{i=0}^{x-1} b_i$, the first test is $Z_1 \le 0$?

The second test is $Z_2 \le 0$

The xth test is $Z_x \le 0$

But $Z_{x+1} = T_0 - \frac{1}{R}\sum_{i=0}^{x} b_i = T_0 - \frac{1}{R}\sum_{i=0}^{x-1} b_i - \frac{b_x}{R} = Z_x - \frac{b_x}{R}$

So each test can update the previous value of Z, as shown in the flowchart of Figure 5. First, at Step 201, T_0 is calculated in accordance with Equation (8), then (Step 202) Z_0 is set equal to T_0. At step 203 a packet counter is reset. Then (204) the first packet (or on subsequent iterations, the next packet) is read from the store 11 and sent to the transmitter 12. At step 205, the control unit computes the value of Z_{n+1}, and the test performed at step 206 as to whether $Z_{n+1} \le 0$. If the test is passed, then it is known that the receiver is safe to begin decoding as soon as it has received this packet. Therefore at step 207 the control unit sends to the transmitter a "start" message to be

sent to the receiver. When the receiver receives this start message, it begins decoding. The packet counter is incremented at 208 and control returns to step 204 where, as soon as the transmitter is ready to accept it, a further packet is read out and transmitted.

The step 201 of calculating T_0 could be done in advance and the values stored. This procedure could of course be adapted, in a similar manner to that previously described, to accommodate different values of R.

It is not essential that this process begin with T_0. One could start with T_1 (in which case the first test is $T_1 \leq 0$?) or, if one chooses always to buffer at least two (or more) packets one could start with T_2, etc.

Although the example given is for encoded video, the same method can be applied to encoded audio or indeed any other material that is to be played in real time.

If desired, in multiple-rate systems, these methods may be used in combination with the rate-switching method described in our international patent application WO04/086721.